Algorithmica (2011) 61:362–388
DOI 10.1007/s00453-010-9404-2

Quadratic Kernelization for Convex Recoloring of Trees

Hans L. Bodlaender · Michael R. Fellows · Michael A. Langston · Mark A. Ragan · Frances A. Rosamond · Mark Weyer

Received: 3 February 2009 / Accepted: 5 March 2010 / Published online: 27 March 2010

Abstract The CONVEX RECOLORING (CR) problem measures how far a tree of characters differs from exhibiting a so-called "perfect phylogeny". For an input consisting of a vertex-colored tree T, the problem is to determine whether recoloring at most k vertices can achieve a convex coloring, meaning by this a coloring where

This research has been supported by the Australian Research Council Centre of Excellence in Bioinformatics, by the U.S. National Science Foundation under grant CCR–0311500, by the U.S. National Institutes of Health under grants 1-P01-DA-015027-01, 5-U01-AA-013512-02 and 1-R01-MH-074460-01, by the U.S. Department of Energy under the EPSCoR Laboratory Partnership Program and by the European Commission under the Sixth Framework Programme. The first author was partially supported by BSIK grant 03018 (BRICKS: Basic Research in Informatics for Creating the Knowledge Society). The second and fifth authors have been supported by a Fellowship to the Institute of Advanced Studies at Durham University, and hosted by a William Best Fellowship to Grey College during the preparation of the paper. A preliminary version of this paper was presented at COCOON 2007 [6].

H.L. Bodlaender (✉)
Department of Information and Computing Sciences, Utrecht University, Utrecht, The Netherlands
e-mail: hansb@cs.uu.nl

M.R. Fellows · F.A. Rosamond
Parameterized Complexity Research Unit, Office of the DVC (Research), University of Newcastle, Callaghan NSW 2308, Australia

M.R. Fellows
e-mail: michael.fellows@newcastle.edu.au

F.A. Rosamond
e-mail: frances.rosamond@newcastle.edu.au

M.R. Fellows · M.A. Langston · M.A. Ragan · F.A. Rosamond
Australian Research Council Centre of Excellence in Bioinformatics, Brisbane, Australia

M.A. Langston
Department of Computer Science, University of Tennessee, Knoxville, TN 37996-3450, USA
e-mail: langston@cs.utk.edu

each color class induces a subtree. The problem was introduced by Moran and Snir (J. Comput. Syst. Sci. 73:1078–1089, 2007; J. Comput. Syst. Sci. 74:850–869, 2008) who showed that CR is NP-hard, and described a search-tree based FPT algorithm with a running time of $O(k(k/\log k)^k n^4)$. The Moran and Snir result did not provide any nontrivial kernelization. In this paper, we show that CR has a kernel of size $O(k^2)$.

Keywords Algorithms · Combinatorial algorithms · Fixed parameter tractability · Kernelization · Convex recoloring · Trees · Phylogeny

1 Introduction

The historically first and most common definition of *fixed-parameter tractability* (or, in short: in FPT) for parameterized problems is solvability in time $O(f(k)n^c)$, where n is the input size, k is the parameter, and c is a constant independent of n and k. Background and motivation for the general subject of parameterized complexity and algorithmics can be found in the books [13, 14, 23]. This basic view of the subject is by now well-known.

Less well-known is the point of view that puts an emphasis on FPT kernelization. For an extensive overview of kernelization, see [17]. Kernelization is central to FPT as shown by the lemma:

Lemma 1 *A parameterized problem* Π *is in FPT if and only if* Π *is decidable and there is a transformation from* Π *to itself, and a function* g, *that reduces an instance* (x, k) *to* (x', k') *such that*:

1. *the transformation runs in time polynomial in* $|(x, k)|$,
2. (x, k) *is a yes-instance of* Π *if and only if* (x', k') *is a yes-instance of* Π,
3. $k' \leq k$, *and*
4. $|(x', k')| \leq g(k)$.

The lemma is trivial, but codifies a shift of perspective. To see how this works, consider the Moran and Snir FPT result [20] for CONVEX RECOLORING, with the running time of $O^*(k/\log k)^k)$. When $n > (k/\log k)^k$, then the Moran and Snir algorithm runs in polynomial time, in fact, in time $O(n^5)$. (So we can run the algorithm and determine the answer, and "transform" the input to either a canonical

M.A. Langston
Computer Science and Mathematics Division, Oak Ridge National Laboratory, Oak Ridge, TN 37831-6164, USA

M.A. Ragan
Institute for Molecular Bioscience, University of Queensland, Brisbane, QLD 4072, Australia
e-mail: m.ragan@imb.uq.edu.au

M. Weyer
Institut für Informatik, Humboldt-Universität zu Berlin, Berlin, Germany
e-mail: mark.weyer@informatik.hu-berlin.de

small NO instance or a canonical small YES instance accordingly.) If $n \leq (k/\log k)^k$ then we simply do nothing; we declare that the input is "already" kernelized to the bound $g(k) = (k/\log k)^k$. If a parameterized problem is FPT, that is, solvable in time $f(k)n^c$, then the lemma above gives us the trivial P-time kernelization bound of $g(k) = f(k)$. Normally for FPT problems, $f(k)$ is some exponential function of k, so the subclasses of FPT

$$Lin(k) \subseteq Poly(k) \subseteq FPT$$

of parameterized problems that admit P-time kernelization to kernels of size bounded by *linear* or *polynomial* functions of k would seem to be severe restrictions of FPT.

It is surprising that so many parameterized problems in FPT belong to these seemingly much more restrictive subclasses. A strong connection to heuristics goes like this: *FPT kernelization* is basically a systematic approach to *polynomial-time* pre-processing. Pre-processing plays a powerful role in practical approaches to solving hard problems. For any parameterized problem in FPT there are now essentially two different algorithm design competitions with independent payoffs:

1. to find an FPT algorithm with the best possible exponential timecost function $f(k)$, and
2. to find the best possible polynomial-time kernelization bound $g(k)$ for the problem.

A classic result on FPT kernelization is the VERTEX COVER $2k$ vertex kernelization due to Nemhauser and Trotter (see [23]). The linear vertex kernelization for PLANAR DOMINATING SET is definitely nontrivial [3]. The question of whether the FEEDBACK VERTEX SET (FVS) problem for undirected graphs has a $Poly(k)$ kernelization was a noted open problem in parameterized algorithmics, only recently solved. The first $Poly(k)$ kernelization for FVS gave a bound of $g(k) = O(k^{11})$ vertices [7]. Improved in stages, the best kernelization bound is now $O(k^2)$ [25].

We prove here a polynomial time kernelization for the CONVEX RECOLORING problem, to a reduced instance that has $O(k^2)$ vertices. The basic problem is defined as follows.

CONVEX RECOLORING
Instance: A tree $F = (V, E)$, a set of colors $\mathcal{C}$, a coloring function $\Gamma : V \to \mathcal{C}$, and a positive integer k.
Parameter: k
Question: Is it possible to modify Γ by changing the color of at most k vertices, so that the modified coloring Γ' is *convex* (meaning that each color class induces a single monochromatic subtree)?

The CONVEX RECOLORING problem is of interest in the context of maximum parsimony approaches to evaluating phylogenetic trees [16, 20]. Some further applications in bioinformatics are also described in [20]; see also [21]. Other applications that have been suggested include:

- Multicast communications in optical wavelength division multiplexing networks [9] where a connected monochrome subgraph represents connecting all the clients in a task group.

- The annotation of biological interaction graphs with, e.g., cellular locality information.

Bar-Yehuda et al. [4] gave a polynomial time $(2+\epsilon)$-approximation algorithm for the CONVEX RECOLORING problem, and also gave an exact algorithm, whose parameterized complexity is $O(n^2 + nk2^k)$. Further improvements on the running time and faster algorithms for weighted variants, including a dynamic programming algorithm that uses linear time for fixed k, were obtained by Ponta et al. [24]. A different proof that the problem can be solved in linear time for fixed k can be found in [6]. The proof in [6] uses a bounded-width tree decomposition of an auxiliary graph and a formulation of the problem in Monadic Second Order logic [10, 11].

Several of the papers cited above consider more general cost models. Here we consider the simplest model, where each recoloring of a vertex has unit cost. It is not hard to generalize our result to the case where we have for each vertex v a positive integer weight $w(v) \in \mathbf{Z}^+$, and recoloring v costs $w(v)$; finding polynomial size kernels for other models is left as a topic for further research.

Instead of working on the CONVEX RECOLORING problem directly, we use an 'annotated' version of the problem, where we allow in addition to fix the color for some vertices. Such an approach (somewhat counterintuitively) is often used in the nascent study of FPT kernelization [3, 17, 18]. For example, annotation is crucial to the linear vertex kernelization algorithms for PLANAR DOMINATING SET [3, 8].

The annotated form of our problem, ANNOTATED CONVEX RECOLORING will be defined formally in Sect. 2, where we also give other preliminary definitions and results. The algorithm is based on a number of kernelization rules. Each rule simplifies the problem instance, either by fixing the color for some vertices, or by removing vertices and edges. It is in general trivial to see that in polynomial time, it can be determined if a rule can be applied, and to apply the rule. First, we give rules that ensure that the number of vertices per color is linear in k; this is done in Sect. 3. Further rules are given in Sect. 4, and these give us, using detailed counting arguments, the kernel of size $O(k^2)$ for ANNOTATED CONVEX RECOLORING. Then, in Sect. 5, we discuss how the results can be transformed to a kernel of size $O(k^2)$ for CONVEX RECOLORING. Some final remarks are made in Sect. 6.

2 Preliminaries

In Sects. 3 and 4, we give a kernelization algorithm for the following generalized problem. In Sect. 5, we show how we can transform back to a kernel for CONVEX RECOLORING.

ANNOTATED CONVEX RECOLORING

Instance: A forest $F = (V, E)$ where the vertex set V is partitioned into two types of vertices, $V = V_0 \cup V_1$, a set of colors $\mathcal{C}$, a coloring function $\Gamma : V \to \mathcal{C}$, and a positive integer k.

Parameter: k

Question: Is it possible to modify Γ by changing the color of at most k vertices in V_1, so that the modified coloring Γ' is *convex*?

A *block* in a colored forest is a maximal set of vertices that induces a monochromatic subtree. For a color $c \in \mathcal{C}$, $\beta(\Gamma, c)$ denotes the number of blocks of color c with respect to the coloring function Γ. A color c is termed a *broken color* if $\beta(\Gamma, c) > 1$, and c is termed an *unbroken color* if $\beta(\Gamma, c) = 1$.

A recoloring Γ' of a coloring Γ is *expanding*, if for each block of Γ', there is at least one vertex in the block that keeps its color, i.e., that has $\Gamma'(v) = \Gamma(v)$. Moran and Snir [20] have shown that there always exists an optimal expanding convex recoloring. It is easy to see that this also holds for the variant where we allow annotations and a forest, as above.

As in [4], we define for each $v \in V$ a set of colors $S(v)$, where $c \in S(v)$ if and only if there are two distinct vertices w, x, $w \neq v$, $x \neq v$, such that w and x have color c (in Γ), and v is on the path from w to x.

We introduce another special form of convex recoloring, called *normalized*. For each vertex $v \in V$, we add a new color c_v. We denote $\mathcal{C}' = \mathcal{C} \cup \{c_v \mid v \in V\}$. A recoloring $\Gamma' : V \to \mathcal{C}'$ of coloring $\Gamma : V \to \mathcal{C}$ is *normalized*, if for each $v \in V$: $\Gamma'(v) \in \{\Gamma(v), c_v\} \cup S(v)$.

Lemma 2 *There is an optimal convex recoloring that is normalized.*

Proof Take an arbitrary optimal convex recoloring Γ'. Define Γ'' as follows. For each $v \in V$, if $\Gamma'(v) \in \{\Gamma(v)\} \cup S(v)$, then set $\Gamma''(v) = \Gamma'(v)$. Otherwise, set $\Gamma''(v) = c_v$. One can show that $\Gamma''(v)$ is convex. Clearly, it is normalized, and recolors at most as many vertices as $\Gamma'(v)$. □

3 Kernelizing to a Linear Number of Vertices Per Color

In the first phase of the kernelization algorithm, rules are repetitively applied, until none is possible, such that the resulting instance is equivalent to the original instance, and such that for each color, there are at most $O(k)$ vertices with that color. To this end, we introduce eight rules. Each rule transforms the current instance to a simpler instance, or returns NO. (Note: instead of returning NO, we could transform to a trivial constant size NO-instance.)

The main ingredients of this phase of the algorithm are the following. For each color c, such that there are at least $2k + 3$ vertices with color c, we first find at least one vertex which receives a fixed color c. Then, Rules 2–5 ensure that there is exactly one vertex v_c with fixed color c and that at most k vertices belong to a different component than v_c. Rules 6 and 7 bound the maximum degree of vertices, and the number of components. Finally, Rule 8 splits colors, and allows us to arrive at a situation with $O(k)$ vertices per color.

We start with a number of lemmas, that lead up to Rule 1. In particular, whenever there is a color c, given to at least $2k + 3$ vertices, then Rule 1 will find a vertex to which we can assign a *fixed* color c.

The next lemma is a well known folklore result concerning trees and forests.

Lemma 3 *Suppose there are α vertices with color c. Then there is a vertex $v \in V$ such that each connected component of $F - v$ contains at most $\frac{\alpha}{2}$ vertices of color c.*

Lemma 4 *Suppose there are $\alpha \geq 2k+2$ vertices with color c. Let v be a vertex such that each connected component of $F - v$ contains at most $\frac{\alpha}{2}$ vertices of color c. Then in any convex recoloring of F with at most k recolored vertices, v has color c.*

Proof Suppose we have $\alpha \geq 2k+3$ vertices with color c. Let v be a vertex such that each connected component of $F - v$ contains at most $\frac{\alpha}{2}$ vertices of color c.

Consider a convex recoloring of F with at most k recolored vertices. Suppose v has a color different from c in this recoloring. Take a vertex $w \neq v$ that had color c in the original coloring. There are at least $\frac{\alpha}{2} \geq k+1$ vertices of color c in the original coloring that do not belong to the component of $F - v$ that contains w. These cannot all have been recolored, so there is at least one vertex x with color c in the recoloring that is in a different connected component of $F - v$ as w. It now follows from the convexity that w must have a color different from c in the recoloring: as v has a color different from c, there cannot be a path of vertices with color c from w to x.

As this holds for each vertex $w \neq v$ that has color c in the original coloring, it implies that we recolored all vertices with color c, a contradiction. Therefore, v must have color c. □

From Lemma 4, we obtain the following rule. The soundness of Rule 1 follows directly from Lemma 4.

Rule 1 *Suppose there are $\alpha \geq 2k+2$ vertices with color c. Let v be a vertex, such that each connected component of $F - v$ contains at most $\frac{\alpha}{2}$ vertices of color c. Assume that $v \notin V_0$, or $\Gamma(v) \neq c$. Now*

- *If v has a color different from c, and $v \in V_0$, then return NO.*
- *If v has a color different from c, and $v \notin V_0$, then set $\Gamma(v) = c$ and decrease k by one.*
- *Fix the color of v: put $v \in V_0$.*

Standard techniques allow us to find in $O(kn)$ time all vertices for which Rule 1 can be applied.

Corollary 1 *When Rule 1 cannot be applied, then for each color c, there are at most $2k+1$ vertices with color c or there is at least one vertex with fixed color c.*

The next three rules limit the number of vertices with the same color as v in each connected component of $F - v$, for a vertex v with a fixed color.

Rule 2 *If there exist $k+1$ vertices $v_1, \ldots, v_{k+1}$ and a vertex $w \in V_0$, such that for all i, $1 \leq i \leq k+1$, v_i belongs to a different component as w and $\Gamma(v_i) = \Gamma(w)$, then return NO.*

Lemma 5 *Rule 2 is sound.*

Proof Each such vertex v must be recolored, and hence we have at least $k+1$ recolorings. □

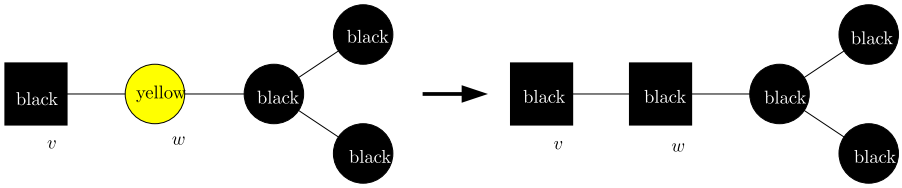

Fig. 1 Example of Rule 3. *Squares* denote vertices with a fixed color. Assume $k = 2$

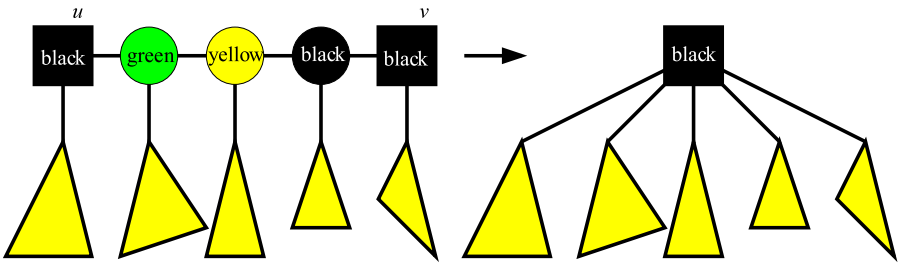

Fig. 2 Example of Rule 4. *Squares* denote vertices with a fixed color

Rule 3 *Let $v \in V_0$ be a vertex of fixed color c. Let W be a component of $F - v$, and suppose $w \in W$ is a neighbor of v. Suppose that W contains at least $k + 1$ vertices with color c, and suppose that $w \notin V_0$ or $\Gamma(w) \neq c$. Then*

- *If w has a color different from c, and $w \in V_0$, then return NO.*
- *If w has a color different from c, and $w \notin V_0$, then give w color c, and decrease k by one.*
- *Fix the color of w: put $w \in V_0$.*

An example is given in Fig. 1.

Lemma 6 *Rule 3 is sound.*

Proof We show that in any convex recoloring which gives v color c, and which recolors at most k vertices, w must have color c. This shows soundness.

Suppose we have a convex recoloring which colors v with c, and that recolors at most k vertices. At least one vertex in W with color c is not recolored, say x. Now, w is on the path from v to x, and v and x have color c, so w must have color c. □

Rule 4 *If there is a tree T in the colored forest that contains two distinct vertices u and v, both of which have fixed color c, then modify T by contracting the path between u and v (resulting in a single vertex of fixed color c). If r denotes the number of vertices of this path that do not have color c, then $k' = k - r$.*

An example of Rule 4 is given in Fig. 2. In this case, k is decreased by two.

Lemma 7 *Rule 4 is sound.*

Proof All vertices on the path from u to v must receive color c in any connected recoloring. So, we must perform r recolorings. Once all vertices on the path have fixed color c, we can obtain an equivalent instance by contracting all these vertices to a single vertex of fixed color c. □

The following rule is obviously sound.

Rule 5 *If there are two distinct trees in the colored forest that both contain a vertex of fixed color c, then return NO.*

Proposition 1 *When Rules 1–5 cannot be applied, then for each color c, there are at most $2k+1$ vertices with color c, or there is exactly one vertex v with fixed color c, and each component of $F-v$ contains at most k vertices with color c.*

Next, we will limit the number of components of $F-v$ containing vertices with the same color as vertex v with fixed color. We first need a definition, and an intermediate result.

Definition 1 Let $v \in V_0$ have fixed color c. A connected component of $F-v$ with vertex set W is said to be *irrelevant*, if both of the following two properties hold:

1. The vertices with color c in W form a connected set that contains a neighbor of v, and
2. For each color $c' \neq c$, if there are vertices with color c' in W, then c' is an unbroken color.

If a component is not irrelevant, it is said to be *relevant*.

Lemma 8 *Let $v \in V_0$ and let W be the vertex set of an irrelevant connected component of $F-v$. Then there is a convex recoloring with a minimum number of recolored vertices that does not recolor a vertex in W.*

Proof Let Γ' be a normalized convex recoloring of Γ with a minimum number of recolored vertices. For each $w \in W$, $S(w) = \{\Gamma(w)\}$, so $\Gamma'(w) \in \{\Gamma(w), c_w\}$. Now, the recoloring Γ'' with for all $x \notin W$, $\Gamma''(x) = \Gamma'(x)$, and for all $w \in W$, $\Gamma''(w) = \Gamma(w)$ is convex, and recolors the same number or less vertices than Γ'. □

Lemma 8 shows that the following rule is sound.

Rule 6 *Let $v \in V_0$. Let W be the vertex set of an irrelevant connected component of $F-v$. Then remove W, and all edges incident to vertices in W, from the forest.*

Rule 7 *Let $v \in V_0$ have fixed color c. If $F-v$ has at least $2k+1$ relevant components, then return NO.*

Lemma 9 *Rule 7 is sound.*

Proof To show this, we introduce a 'currency' with which we pay recolorings of vertices: a *har* is 'half-a-recoloring'. For a convex recoloring, we assign at most two hars to vertices. Suppose we recolor a vertex x from color c to color c'. Then one har is assigned to x, and one har is assigned to an arbitrary vertex which has color c' in the recoloring. (If no such vertex exists, the second har is not assigned.)

Suppose we have a convex recoloring with at most k recolored vertices. There must be a component of $F - v$ for which no vertex has assigned to it a har. Let W be the vertex set of this component. As the component was relevant, one of the following cases must hold.

- There is a vertex $w \in W$ with color c, and the vertices with color c do not form a connected set. Take two vertices with color c in W, say w and x in different blocks. All vertices on the path from w to x belong to W. Either w, x, or a vertex with a color different from c on the path from w to x must be recolored, and thus a har is assigned to W, yielding a contradiction.
- The vertices with color c in W form a connected set, but do not contain a neighbor of v. Similar to the previous case, all vertices in W with color c must be recolored, or all vertices on the path from v to vertices in W must get color c; in all cases, a har is assigned to a vertex in W.
- There is a vertex $w \in W$ with broken color $c' \neq c$. If w is recolored, it gets a har, contradiction. Otherwise, the block of color c' containing w in the recoloring is a subset of W (as v has a fixed color $\neq c'$). Other vertices with color c' must have been recolored, and thus, a har is given to a vertex in W with color c'.

As each case yields a contradiction, there does not exist a convex recoloring of F with at most k recolored vertices. □

We can observe that an instance that is reduced with respect to Rules 1–7 has $O(k^2)$ vertices per color: if there are more than $2k + 2$ vertices of color c, there is a vertex v with fixed color c, and each of the at most $2k$ components of $F - v$ can contain at most k vertices of color c: if a component of $F - v$ contains more than k vertices of color c, then Rule 3, possibly followed by Rule 4 will either decrease the number of vertices or return NO. A further reduction of the number of vertices per color can be used by introducing new colors and vertices. While this step increases the size of the instance, it adds more structure and thus will enable further rules that again decrease the size of the instance.

Rule 8 *Let $v \in V_0$ be a vertex with fixed color c. Let W be the vertex set of a relevant component of $F - v$ that contains vertices with color c. Suppose there are vertices with color c, not in $\{v\} \cup W$. Then*

- *Take a new color c', not yet used, and add it to $\mathcal{C}$.*
- *Take a new vertex v', and add it to F.*
- *Set $v' \in V_0$. Color v' with c'.*
- *For all $w \in W$ with color c, give w color c'.* (*Note that k is* not *changed.*)
- *For the edge $\{v, w\}$ with $w \in W$: remove this edge from F and add the edge $\{v', w\}$.*

So, basically we 'split' v into two copies; one with neighbors in W, and one with its other neighbors. For W and the second copy of v, v', we replace the color c by

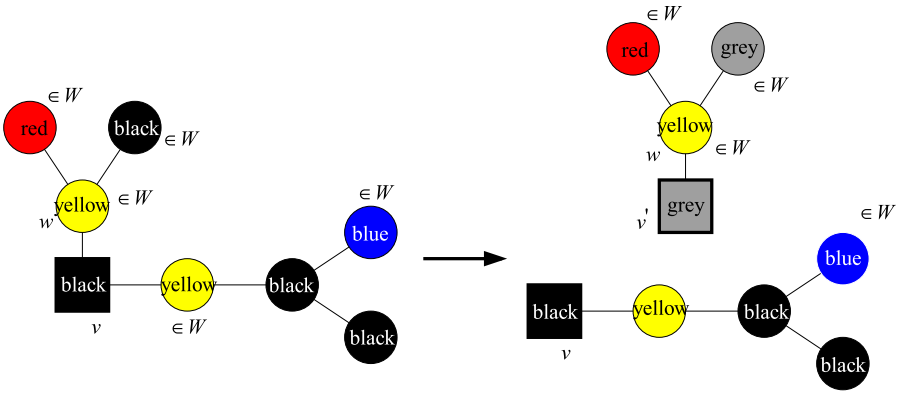

Fig. 3 Example of Rule 8. *Grey* is a new color

a new color. The idea is that the recolorings in the two parts do not influence each other with respect to what happens with color c, as v is a fixed vertex with color c separating these parts. Note that if W is in a different subtree of forest F than v, the new vertex v' is isolated. See for an example Fig. 3.

Lemma 10 *Rule 8 is sound.*

Proof Let F be the forest before applying the rule, and F' the forest after the rule.

Suppose we have a convex recoloring of F with at most k recolored vertices. Now, recolor the vertices in F' as follows: if a vertex gets a color $c'' \neq c$ in F, then recolor it in the same way in F'. If w gets color c, then color it with color c if $w \notin W$, and color it with color c' if $w \in W$. It is not hard to check that this gives a convex coloring, again with at most k recolored vertices.

Suppose we have a convex recoloring of F' with at most k recolored vertices. Now, recolor all vertices in F as in this recoloring, but give all vertices with color c or c' in F' color c in F. This gives again a convex recoloring. □

The algorithm works as follows. Rules 1–7 are applied in any order, till none is possible. Then, we apply Rule 8 for each vertex v with a fixed color as long as $F - v$ has at least two relevant components. This clearly gives a polynomial time algorithm: each of the Rules 1–7 either decreases the number of edges, or increases the number of vertices with a fixed color while not increasing the number of edges. Rule 8 decreases the sum over all vertices $v \in V_0$ of the number of relevant components of $F - v$ minus 1. Determining whether a rule can be applied, and applying it, can clearly be done in polynomial time.

Theorem 1 *An instance that is reduced with respect to Rules 1–8 has at most* $2k + 1$ *vertices per color.*

Proof Consider a color c. If there is no vertex with fixed color c, then there are at most $2k + 1$ vertices with color c, otherwise Rule 1 applies. If there is a vertex v

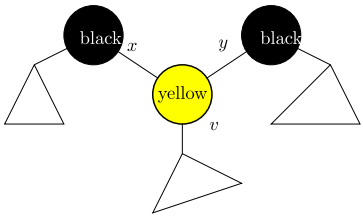

Fig. 4 A gluing vertex

with fixed color c, then at most one component of $F - v$ contains vertices with color c, otherwise, Rule 8 applies and decreases the number of vertices with color c. This component contains at most k vertices with color c (Rules 2, 3, and 4), so in this case, there are at most $k + 1$ vertices with color c. □

4 Kernelizing to a Quadratic Number of Vertices

In this section, we work in a series of steps towards a kernel with $O(k^2)$ vertices. A number of new structural notions are introduced. In particular, we distinguish several types of unbroken colors.

We start with giving some high level intuition behind our methods. The analysis and rules are based on a distinction between different types and subtypes of colors. We have two types of colors: unbroken colors (for each unbroken color c, the vertices with color c induce a subtree) and broken colors. First, a relatively easy argument shows how to obtain a bound of $O(k^2)$ on the number of vertices with a broken color: we linearly bound the number of broken colors, and by Theorem 1 there are a linear number of vertices per broken color. The remainder of the section is then devoted to bounding the number of vertices with an unbroken color. In some optimal convex recolorings, we may need to recolor vertices with an unbroken color, namely to connect two blocks of a broken color. We distinguish different types of unbroken colors, depending how they can be used to "bridge" any two blocks of broken colors. Some unbroken colors are called irrelevant: we know that they will not be used to make such a bridge. All vertices with irrelevant colors will be removed. The remaining unbroken colors are partitioned into glue-containing broken colors and stitch-containing broken colors. A glue-containing broken color contains a gluing vertex: a gluing vertex is incident to two blocks of the same broken color, and thus can unify these blocks when recolored. A stitch-containing broken color does not contain gluing vertices, but contains vertices on a stitch; a stitch is a path of length at least two between two blocks with the same broken color of 'cost' at most k. The cost-measure is a technical notion, giving a lower bound on the number of recolorings needed if the bridge is made via this path, and thus bridges with cost more than k will never used. In addition, the cost bound helps to bound the number of vertices with a stitching color. So, glue-containing colors form essentially a short (length 1) bridge between two blocks of a broken color, and stitch-containing colors form a longer bridge but with small cost. See also Figs. 4 and 7.

4.1 Broken Colors

In several of the counting arguments in Sect. 4, we use the notion of half-a-recoloring ('har'), that was used in the proof of Lemma 9. Here, we assign hars to colors. If we recolor a vertex, we assign one har to its old color, and one har to its new color. As a broken color that does not get a har assigned to it remains broken, we have that the following rule is sound.

Rule 9 *If there are more than* $2k$ *broken colors, then return NO.*

From Rule 9 and Theorem 1, we directly have:

Corollary 2 *After Rules* 1–9 *are applied, we have* $O(k^2)$ *vertices with a broken color.*

Thus, our task is now to obtain rules that bound the total number of vertices with an unbroken color. These vertices and their colors can play different roles in the instance, and thus we distinguish different types. In particular, we look at three types of colors: glue-containing, stitch-containing, and irrelevant.

4.2 Gluing Vertices and Glue-Containing Colors

Here, we define the notion of gluing vertices and glue-containing colors.

Definition 2 A vertex v is *gluing*, if

- it has two neighbors x and y, such that the color of x equals the color of y, and the color of x and y is different from the color of v, and
- the color of v is unbroken

Moreover, we say that v is *gluing for* the color of x and y. A color c is *glue-containing*, if there is at least one vertex with color c that is gluing.

See Fig. 4. The yellow vertex is a gluing vertex; it is gluing for black. White thus is a glue-containing color. Note that glue-containing colors are always unbroken. In this subsection, we introduce rules that bound the number of vertices with a glue-containing color to $O(k^2)$. In the successive subsections, we will deal with the other subcategories of the unbroken colors.

Note that if v is gluing for color c, then v separates (at least) two blocks of color c: x and y belong to different blocks. When we recolor v with color c, then these blocks become one block (are *glued* together).

For a color c, let W_c be the set of vertices that have color c or are gluing for color c. A *bunch* of vertices of color c is a maximal connected set of vertices in W_c, that is, a maximal connected set of vertices that either have color c or are gluing for color c.

In other words, *bunches* of color c are the *blocks* of color c, when we recolor all vertices, gluing for color c with c. An example can be found in Fig. 5. Next, we will bound the number of gluing vertices, and thus the number of vertices with a glue-containing color.

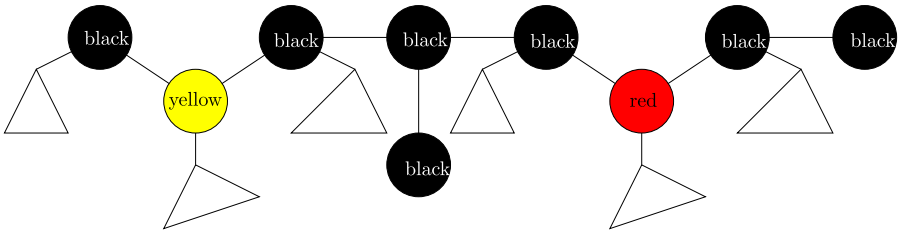

Fig. 5 (Color online) A bunch. All *black vertices* belong to the same bunch. The *yellow* and *red vertex* are gluing for *black*. *Yellow* and *red* are glue-containing colors

Lemma 11 *Suppose W is a bunch of color c that contains ℓ vertices that are gluing for color c. Then in any convex recoloring, at least ℓ vertices in W are recolored, and for each, either the old, or the new color is c.*

Proof First, note that the bunch contains at least $\ell + 1$ blocks of vertices of color c. Suppose we recolor less that ℓ vertices in W. Then, there is at least one block in W whose vertices are not recolored. Take an arbitrary vertex v_r from that block, and consider the rooted tree with vertex set W and v_r as root. Now, each gluing vertex v for color c in the bunch has a child w with color c. So, in a convex recoloring, we must either give v color c, or give w a color different from c. This gives at least ℓ recolored vertices. □

Recall that whenever we recolor a vertex with color c to color c', we assign a har to c and a har to c'. Thus, a bunch with color c with ℓ gluing vertices implies that ℓ hars are assigned to color c. As a consequence, we have the soundness of the following rule.

Rule 10 *If there are more than $2k$ vertices that are gluing, then return NO.*

Corollary 3 *There are $O(k)$ glue-containing colors, and $O(k^2)$ vertices with a glue-containing color.*

Bounding the number of bunches As a preparation for the next two subsections, we introduce a rule that bounds the total number of bunches. We need the following lemma.

Lemma 12 *Suppose there are ℓ bunches with color c. Then the number of recolored vertices whose old or new color equals c, is at least $\ell - 1$.*

Proof Suppose bunches $B_1, \ldots, B_\ell$ have color c. If each of these bunches contains a recolored vertex whose old or new color equals c, then the total number of such recolored vertices is ℓ and we are done. Therefore, we can assume there is a bunch with color c, say B_1 that does not contain a recolored vertex with old or new color c. So, B_1 contains vertices with color c that are not recolored.

For each i, $2 \leq i \leq \ell$, consider bunch B_i. If B_i belongs to a different tree from B_1, then at least one vertex in B_i whose original color was c must be recolored. Suppose

B_i belongs to the same tree as B_1 and that B_i does not have a recolored vertex whose old or new color is c. Let w be the first vertex outside B_i on the path from B_i to B_1. As both B_1 and B_i contain vertices with color c that are not recolored, w must have new color c. We associate w to B_i. By the definition of bunch, w must have an old color, different from c, and w cannot be associated to another bunch. So the lemma follows. □

So, a color with ℓ bunches gets at least $\ell - 1$ hars assigned to it. Thus, the soundness of the following rule follows.

Rule 11 *Suppose that for each broken color c, there are ℓ_c bunches. If the sum over all broken colors c of $\ell_c - 1$ is at least $2k + 1$, then return NO.*

4.3 Stitches

In this section, we introduce the notions of *stitching vertices*, *stitches*, and *stitch-containing colors*. All colors that are unbroken, but not glue-containing or stitch-containing are *irrelevant*. We will give a rule that enables us to remove all vertices with an irrelevant color, and prove its correctness. The definition of a *stitching vertex* is complicated, but this seemed necessary in order to arrive at a kernel of quadratic size.

We first define the *cost* of a path between two vertices with the same color that belong to different bunches. Let c be a broken color, and let v and w be two vertices with color c, in the same subtree of F, but in different bunches of color c. The *cost* of the path P from v to w consists of two parts. The cost is the sum of these two parts. The first part is simply the number of vertices on P that do not have color c. The second part equals $\sum_{c'} s(c')$, where we sum over all colors $c' \neq c$ such that

1. c' is unbroken and c' is not glue-containing and
2. there exists a vertex on P with color c'.

The value of c' is determined as follows: delete the vertices of the path P and consider the blocks with color c' after this deletion. If one of these blocks contains a vertex of fixed color (c'), then $s(c')$ is the number of vertices in the remaining blocks of color c'. Otherwise, $s(c')$ is the number of vertices in all blocks with color c' except the one with the largest number of blocks.

An example of the cost of a path is given in Fig. 6. The path P from v to w has cost 9. The first part of the cost measure is 4, as the path has 4 non-black vertices. For the second part, yellow contributes 2 to the cost (count all yellow vertices not on P and not in the largest block if we remove P); red contributes 3 (count all vertices not on P except the yellow vertex with fixed color), and blue contributes 0. If $k \geq 9$, then P is a stitch. If yellow, blue, and red are unbroken and not glue-containing (in the remainder of the forest), then they are stitch-containing.

The following proposition also shows the intuition behind the cost measure.

Proposition 2 *Let v and w be vertices of the same broken color c that belong to the same subtree of F, but belong to different bunches of color c. Suppose the cost of the*

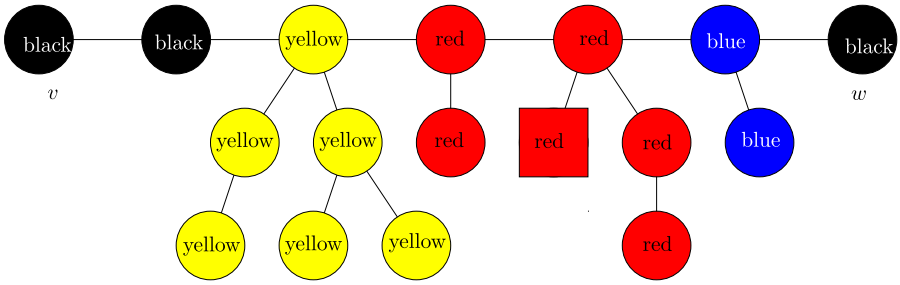

Fig. 6 (Color online) A possible stitch. The cost of the path from v to w is 9: 4 for the vertices on the path, and 2 for *yellow*, 3 for *red*, and 0 for *blue*. If $k \geq 9$, then this path is a stitch, and *yellow*, *red*, and *blue* can be stitch-containing colors

path from v to w is α. Suppose we have a convex recoloring, such that all vertices on that path from v to w have color c. Then, at least α vertices are recolored.

Proof Each vertex with a color unequal to c on the path is recolored, and counted in the first part of the sum. Consider an unbroken color $c' \neq c$ given to a vertex on the path. When all vertices on the path are recolored with c, it is possible that c' is no longer unbroken. Consider the forest F', obtained by removing all vertices on the path from v to w. At most one component of F' can contain vertices with color c' that are not recolored. Thus, if there is a vertex $v_{c'}$ with fixed color c', we must recolor all vertices with color c' in the components not containing $v_{c'}$; otherwise, we must recolor all of the vertices with color c' in every component except for one. □

A *stitch* of color c is a path P between two vertices v, w, both of color c, such that:

1. v and w are in different bunches,
2. all vertices except v and w have a color different from c, and do not belong to V_0, and
3. the cost of P is at most k.

A vertex v is *stitching* for color c, if:

1. v is not gluing for color c,
2. v is on a stitch of color c, and
3. the color of v is different from c,

A vertex is *stitching*, if it is stitching for at least one color. (Note that this vertex will be stitching for a broken color.) A color c is *stitch-containing* if there is at least one vertex with color c that is stitching, and c is unbroken and not glue-containing.

A color is *irrelevant*, if it is neither broken, glue-containing, nor stitch-containing. Summarizing, we have the following types of vertices:

1. vertices with a broken color,
2. vertices with a glue-containing color,
3. vertices with a stitch-containing color that belong to a stitch,

4. vertices with a stitch-containing color that do not belong to a stitch—these will be partitioned into pieces in Sect. 4.5, and
5. vertices with an irrelevant color.

Lemma 13 *Suppose there is a convex recoloring with at most k recolored vertices. Then there is an optimal convex recoloring that is normalized such that for each broken color c, all vertices that receive color c have color c in the original coloring or are gluing for c or are stitching for c.*

Proof Suppose we have an optimal convex recoloring Γ' with at most k recolored vertices. We assume that Γ' is normalized (cf. Lemma 2). From all such possible Γ', take one with a maximum number of vertices v with $\Gamma'(v) = c_v$.

Consider a broken color c. Let V_c be the set of vertices that have color c and are not recolored, i.e., $V_c = \{v \mid \Gamma'(v) = \Gamma(v) = c\}$.

Claim Each vertex with color c in the recoloring Γ' belongs to V_c or is on a path between two vertices in V_c; all vertices on this path have color c in the recoloring.

Proof Suppose not. Take a vertex v that is recolored, that received color c, and that is not on a path between two vertices in V_c. Let W be the set of vertices w in F, with the property that w has color c in the recoloring, and each path from w to a vertex in V_c passes through v. By assumption, we have that $v \in W$. Now, if we recolor all vertices in W by a color of the form c_w, i.e., change Γ' by setting for each $w \in W$ the color of w to c_w, we obtain a recoloring with the same number of recolored vertices as Γ'. This recoloring is also convex, by the construction of W, but has a larger number of vertices that are mapped to their color c_w, a contradiction. □

Now consider a vertex v which has color c in Γ'. Suppose v does not have color c in Γ. Then, by the previous claim, there must be two vertices w and $x \in V_c$ such that v is on the path from w to x.

Consider the path from w to x. Let w' be the last vertex in V_c on this path before v, and let x' be the first vertex in V_c on this path after v. Consider the path from w' to x'. If w' and x' are in the same bunch, then v is the only vertex between w' and x' on the path, i.e., v is gluing for c. So, we can assume that w' and x' belong to different bunches.

By Proposition 2, the cost of the path from w' to x' must be at most k (otherwise, we recolor too many vertices). Thus, this path is a stitch, and therefore v is stitching for c. □

4.4 Irrelevant Colors

At this point, we are able to deal with the irrelevant colors, and prove the soundness of rules that remove all vertices with an irrelevant color.

Lemma 14 *Suppose there is a convex recoloring with at most k recolored vertices. Then there is an optimal convex recoloring such that no vertex with an irrelevant color is recolored.*

Proof Again, assume we have a normalized recoloring Γ' with at most k recolored vertices. Using the construction of the previous lemma, we may assume that for each broken color c, all vertices with color c in the recoloring, have color c in the original color, or are gluing or stitching for c.

Consider an irrelevant color c'. As no vertex with color c' is on a stitch, we have, by the previous lemma, that no vertex with color c' is recolored to a broken color. As the recoloring is normalized, all recolored vertices v with color c' receive the color c_v.

Suppose we recolor at least one vertex with color c'. Now construct a recoloring Γ'' as follows: for all vertices v, with $\Gamma(v) \neq c'$, set $\Gamma''(v) = \Gamma'(v)$, and for all v with $\Gamma(v) = c'$, set $\Gamma''(v) = c'$. Γ'' is convex: all colors $c'' \neq c$ are given to the same vertices in Γ' and in Γ'', and all vertices with $\Gamma(v) = c'$ keep color c'; as c' is irrelevant, it is unbroken. Thus, each color is given to exactly one block. However, Γ'' recolors fewer vertices than Γ' does, contradiction. □

From Lemma 14, it follows that we can fix the color of vertices with an irrelevant color, and thus that the following rule is sound.

Rule 12 *Let $v \in V_1$ be a vertex with an irrelevant color. Put v into V_0 (i.e., fix the color of v).*

The following rule, in combination with earlier rules, allows us to obtain a reduced instance having no vertices with an irrelevant color.

Rule 13 *Let $v \in V_0$ be the only vertex with some color c. Then remove v and its incident edges.*

Soundness is easy to see: as v has a fixed color, the same recolorings in the graph with and without v are convex. The combination of Rules 4, 12, and 13 causes the deletion of all vertices with an irrelevant color: all such vertices first get a fixed color, then all vertices with the same irrelevant color are contracted to one vertex, and then this vertex is deleted. So, we can also use instead the following rule.

Rule 14 *Suppose c is an irrelevant color. Then remove all vertices with color c.*

4.5 Pieces of a Stitch-Containing Color

At this point, we have $O(k^2)$ vertices with a broken or with a glue-containing color, and no vertices with an irrelevant color. Thus, we need to focus on bounding the number of vertices with a stitch-containing color. In order to reduce the number of such vertices we introduce the concept of a *piece of color c*.

Consider a stitch-containing color c. Recall that c is unbroken. Consider the subtree T_c of the forest, induced by the vertices with color c. Now, let F_c be the forest, obtained by removing from T_c all vertices that are on a stitch. The vertex sets of the components of F_c are the *pieces*. In other words, a *piece of color c* is a maximal subtree of vertices with color c that do not belong to a stitch. An example is given in

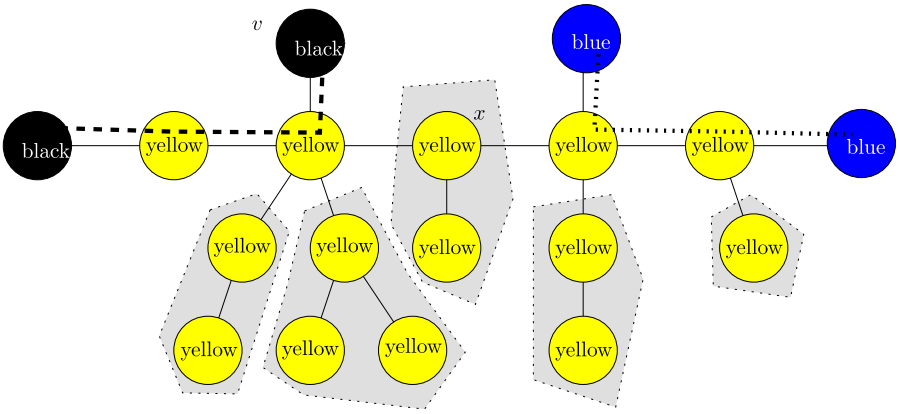

Fig. 7 (Color online) Pieces and stitches. The *dotted* and *dashed line* mark stitches; one for *black*, and one for *blue*. The *gray areas* are pieces. *Yellow* is stitch-containing

Fig. 7. Note that v is on a path between two stitches, but does not belong to a stitch itself, and thus belongs to a piece. Assume that we have exhaustively applied all of the rules described so far, and therefore we do not have vertices with an irrelevant color.

First, we deal with pieces that contain vertices with a fixed color.

Lemma 15 *Suppose W is the vertex set of a piece of stitch-containing color c. Suppose there is a vertex $v \in W \cap V_0$. Then if there is a convex recoloring with at most k recolored vertices, then there is a convex recoloring that does not recolor any vertex in W.*

Proof Suppose there is a convex recoloring Γ' with at most k recolored vertices. We can assume that there are no vertices with an irrelevant color, that Γ' is normalized, and that for each broken color c, the vertices v with $\Gamma'(v) = c$, either have $\Gamma(v) = c$, or are gluing or stitching for c. We also assume Γ' recolors a minimum number of vertices.

Suppose now that W is the vertex set of piece of stitch-containing color c, and that $v \in W \cap V_0$. We claim that for all $w \in W$, $\Gamma'(w) = \Gamma(w) = c$. Suppose not. Take a vertex $w \in W$ with $\Gamma'(w) \neq \Gamma(w) = c$. Consider the path from w to v. Let x be the last vertex on this path with $\Gamma'(x) \neq c$. As $v \in V_0$, $\Gamma'(v) = c$, so $x \neq v$, and x is adjacent to a vertex y with $\Gamma'(y) = c$.

As x belongs to a piece, it is not stitching or gluing, and hence we have that $\Gamma'(x) = c_x$. Thus, if we change the color of x in Γ' to c, we still have a convex recoloring (c still has one block, and there are no other vertices with the same color as x in Γ'). Hence, there is a convex recoloring with fewer recolored vertices than Γ', a contradiction. Thus, for all $w \in W$, $\Gamma'(w) = \Gamma(w)$. □

Lemma 15 shows directly that the following rule is sound.

Rule 15 *Let W be a vertex set of a piece of stitch-containing color c that contains at least one vertex in V_0. Then put all of the vertices of W into V_0.*

As a result of this rule and Rule 4, a piece containing a vertex with a fixed color will contain only one vertex.

Our next and last rule helps to bound the number of vertices in the largest piece for a stitch-containing color for colors where there is no vertex this color fixed. This is useful, as this number was not counted in the cost of a stitch. We first give the rule, and then prove its soundness.

Rule 16 *Suppose c is a stitch-containing color, and there are α vertices with color c. Suppose there is no vertex in V_0 with color c. If W is the vertex set of a piece of color c, and $|W| > \alpha/2$, then put all of the vertices of W into V_0.*

Lemma 16 *Rule 16 is sound.*

Proof We will show that if there is a convex recoloring with at most k recolored vertices, then there is one that does not recolor any vertex in W. Soundness then directly follows.

Let c, α, W be as in the rule. Suppose there is a convex recoloring Γ' with at most k recolored vertices. As before, we can assume that there are no vertices with an irrelevant color, that Γ' is normalized, and that for each broken color c, the vertices v with $\Gamma'(v) = c$, either have $\Gamma(v) = c$, or are gluing or stitching for c. We also assume Γ' recolors a minimum number of vertices.

Let $Q = \{v \in V \mid \Gamma(v) = c\}$ be the vertices with color c. Let X be the vertices in Q that are recolored, and $Y = Q - X$ be the vertices in Q that are not recolored. Let $Z \subseteq X$ be the vertices in X that are recolored to a broken color. Note that vertices in Z must be stitching for some broken color: vertices recolored to a broken color are either gluing or stitching to a broken color, but as c is stitch-containing, it is not glue-containing and thus vertices in Z are not gluing. The vertices in $X - Z$ are recolored with new colors of the form c_v.

As c is stitch-containing (and hence unbroken), Q induces a subtree. Each tree in the forest, obtained by removing the vertices of Z from Q is called a *superpiece*. Each piece of color c is contained in a superpiece: a piece does not contain stitching vertices, and hence contains no vertices recolored to a broken color. An example is given in Fig. 8.

In particular, the piece with vertex set W is contained in a superpiece, say with vertex set W'. Vertices v in superpieces are not recolored to broken colors, so either are not recolored or recolored to new color c_v, using that Γ' is normalized.

Now, change Γ' to a recoloring Γ'' in the following way: for all $v \in V - Q$ and for all $v \in Z$, set $\Gamma''(v) = \Gamma'(v)$. For all $v \in Q - Z - W'$, set $\Gamma''(v) = c_v$, and for all $v \in W'$, set $\Gamma''(v) = c = \Gamma(v)$. In other words, we recolor all vertices in superpieces, except W' to the color of the form c_v, and do not recolor the vertices in the superpiece W'.

Γ'' is convex. As all vertices v in W' have $\Gamma'(v) \in \{c, c_v\}$, and c_v is a color that can only be given to v, the change from Γ' to Γ'' cannot increase the number of blocks for any color $\neq c$. Also, as W' induces a subtree, there is only one block with color c.

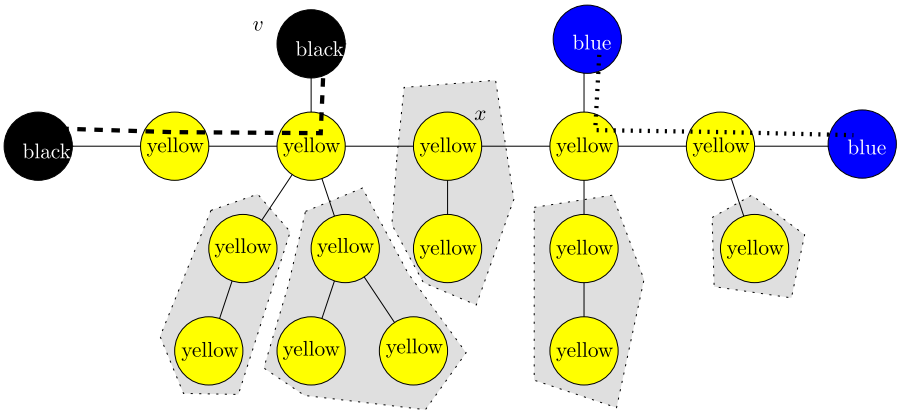

Fig. 8 Pieces and superpieces. A convex recoloring of the tree of Fig. 7 is given, and pieces and superpieces are shown with *gray areas*

Γ'' recolors at most as many vertices as Γ'. To show this, we compare the number of vertices in Q which are recolored: note that if $\Gamma(v) \neq c$, then $\Gamma'(v) = \Gamma''(v)$, so we only need to analyze which vertices in Q are recolored. We consider two cases.

The first case is that there is a vertex v in W' which is not recolored by Γ'. Then, any vertex $w \in Q$ that is recolored by Γ'' either is recolored to a broken color (i.e., it belongs to Z and $\Gamma''(w) = \Gamma'(w)$), or belongs to another superpiece. In the latter case, the path from v to w contains a vertex z in Z, and now $\Gamma'(z) \neq c$ implies that $\Gamma'(w) \neq c$. So, all vertices recolored by Γ'' are also recolored by Γ'.

In the second case, all vertices in W' are recolored by Γ'. The number of vertices in Q, recolored by Γ' is hence at least $|W'| \geq |W| > \alpha/2$. The number of vertices in Q, recolored by Γ'' equals $|Q - W'| = |Q| - |W'| < \alpha - \alpha/2$. So Γ' recolors more vertices that Γ'', contradicting the minimality of Γ'.

As Γ'' is a convex recoloring which recolors at most as many vertices as Γ', and does not recolor any vertex in $W' \supseteq W$, the result follows. □

If there is a large piece, found by Rule 16, then it will be contracted to a single vertex by Rule 4.

Corollary 4 *Suppose none of the Rules* 1–16 *can be applied. Let c be a stitch-containing color, and suppose α vertices have color c. Then, for each piece of color c, one of the following two cases holds*:

- *It consists of one vertex with fixed color c*
- *It contains at most $\alpha/2$ vertices with color c.*

The description of the algorithm is now done: we apply all rules, until none is possible anymore. One easily verifies that we obtain in polynomial time an equivalent instance. To show that this instance has $O(k^2)$ vertices we still need some complicated counting arguments.

4.6 Kernel Size

In this subsection, we show that the size of the resulting kernel is indeed $O(k^2)$. For this, we need to bound the number of vertices with a stitch-containing color. We do this by tagging some of these vertices, and then counting first the number of tagged vertices and then the number of untagged vertices with a stitch-containing color. This tagging procedure is not needed for the algorithm, but only to show the bound of the kernel size. Each tag is labeled with a broken color. Basically, when we have a stitch for a broken color c, we tag the vertices that count for the cost of the stitch with color c.

Tagging is done as follows. For each stitch for broken color c, and for each stitch-containing color c' with a vertex on the stitch with color c', consider the blocks of color c' obtained by removing the vertices on the stitch. Let Q be the set of vertices in these blocks. If there is no vertex with color c' in V_0, then take a block with vertex set W such that the number of vertices in this piece is at least as large as the number of vertices in any other piece. Tag all vertices in $Q - W$ with color c. If there is a vertex v with color c' in V_0, then tag all vertices with color c, except those that are in the same block as v.

Comparing the tagging procedure with the definition of the cost of a stitch, we directly note that the number of vertices tagged equals the cost of the stitch. In particular, this implies that for each stitch of broken color c, we tag at most k vertices with c.

Counting the number of tagged vertices We now give a bound on the number of vertices with a tag; these form a subset of the vertices with a stitch-containing color. To do this, we consider a broken color c, and count the number of vertices with a stitch-containing color that are tagged with c.

Lemma 17 *Let c be a broken color with ℓ_c bunches. A reduced instance has at most $2k(\ell_c - 1)$ vertices that are tagged with c.*

Proof If there is only one bunch with color c, then there are no stitches for c, and hence there are no vertices tagged with c. So, in the remainder of the proof, we assume $\ell_c \geq 2$.

We define the *bunch-stitch* graph for color c as the graph, formed as follows. Take F. Contract each bunch of color c to a single vertex. Call these *bunch vertices*. Now, remove all vertices and incident edges that are not either a bunch vertex or a vertex that is stitching for color c. Clearly, the bunch graph is a forest.

A *branch vertex* is a stitching vertex in the bunch-stitch graph of degree at least three. As each leaf of the bunch-stitch graph is a bunch vertex, there are at most ℓ_c leaves in the bunch-stitch graph, and hence there are at most $\ell_c - 1$ branch vertices. A path in the bunch-stitch graph such that both endpoints are a bunch vertex or a branch vertex, and all inner vertices are stitching vertices of degree two in the bunch-stitch graph is called a *thread*.

Note that there are at most $2\ell_c - 2$ threads. This can be seen as follows. If we contract each stitching vertex of degree two with a neighbor in the bunch-stitch graph,

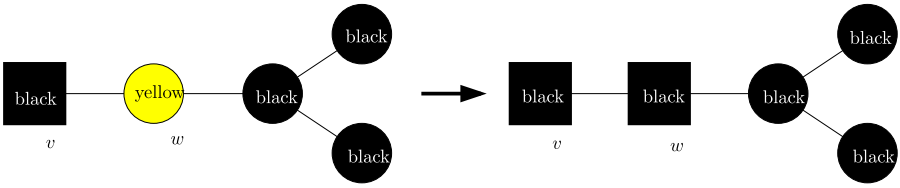

Fig. 9 A tree and its bunch-stitch graph for black. There are three threads, marked by different *dotted lines*

we obtain a forest where each edge corresponds to a thread. The vertices of the forest are the bunch and branch vertices, and hence there are at most $2\ell_c - 1$ vertices, so at most $2\ell_c - 2$ edges and hence threads.

We now view threads as paths in F, by replacing bunch vertices by a vertex in the corresponding bunch (adjacent to the next stitching vertex in the thread). We extend the cost measure, defined in the beginning of Sect. 4.3 to threads. See Fig. 9. We give a tree, and the corresponding bunch-stitch graph for black. We have three stitches. The left stitch (dashed line) has cost 2 for two red vertices on the stitch. The right stitch (dotted line) has cost 4: 2 for two red vertices on the stitch, and 2 for the two smaller components. The lower stitch has cost 3: a red and a blue vertex on the stitch and one vertex in a component.

For each thread, the cost is at most k, as each thread is contained in a stitch. So, the total cost of all threads is at most $2k(\ell_c - 1)$.

Finally, note that each vertex that is tagged with color c counts for the cost of at least one of the threads. So, the number of vertices tagged with c is at most the sum of the costs of the threads, and hence at most $2k(\ell_c - 1)$. □

Corollary 5 *There are at most* $4k^2$ *tagged vertices with a stitch-containing color.*

Proof Again write ℓ_c for the number of bunches of a broken color c. As the sum over all broken colors of the term $\ell_c - 1$, is at most $2k$, there are at most $4k^2$ vertices with a stitch-containing color that are tagged. □

Untagged vertices with a stitch-containing color A simpler proof suffices to count the number of vertices with a stitch-containing color that are not tagged.

Lemma 18 *Let c be a stitch-containing color. There is at most one piece of color c that contains a vertex that is not tagged with a broken color.*

Proof Suppose v has stitch-containing color c and is not tagged with a broken color. As all vertices on a stitch are tagged, v belongs to a piece of color c. Consider a vertex w in another piece. By the definition of piece, there is a vertex on a stitch on the path from v to w. As this stitch does not tag v, w is tagged by the stitch. □

Corollary 6 *For each stitch-containing color, the number of untagged vertices is at most the number of tagged vertices or* 1.

Proof By Lemma 18, there is only one piece with untagged vertices, and by Corollary 4, it contains at most half of all vertices with the color. □

Corollary 7 *In a reduced instance, there are at most* $8k^2$ *vertices with a stitching color.*

Kernel size We can now combine all of the steps in our argument so far, and show the conclude the main result of the paper.

Theorem 2 *In time polynomial in* n *and* k*, a kernel for* ANNOTATED CONVEX RECOLORING *can be found with at most* $16k^2 + 4k$ *vertices.*

Proof With standard techniques, one can observe that all rules can be carried out in time polynomial in n and k.

In the reduced instance, there are at most $2k$ broken colors, and at most $2k$ glue-containing colors. For each of these colors, there are at most $2k + 1$ vertices with that color. So, in total at most $8k^2 + 4k$ vertices have a broken or glue-containing color. There are at most $8k^2$ vertices with a stitch-containing color, and no vertices with an irrelevant color, so in total we have at most $16k^2 + 4k$ vertices. □

5 A Quadratic Kernel for (Unannotated) Convex Recoloring

In the previous section, we have seen how to obtain a kernel with $O(k^2)$ vertices for the ANNOTATED CONVEX RECOLORING problem. In this section, we discuss how this can be modified to a kernel for CONVEX RECOLORING, i.e., with some final steps, we can ensure that we do not have annotations (vertices with a fixed color), and that we have a tree (instead of a forest).

Lemma 19 *Let* $v \in V_1$ *be a vertex which has at least* $k + 1$ *neighbors with the same color. Then,* v *is not recolored in a convex recoloring with at most* k *recolored vertices.*

Proof Suppose v is recolored. Then, at least two neighbors w_1, w_2 of v are not recolored, but now the resulting coloring is not convex, as the path from w_1 to w_2 uses in the recoloring a vertex with a color different from the color of w_1 and w_2. This is a contradiction. □

With Lemma 19, we can easily undo annotations: we add to each vertex $v \in V_0$, $k + 1$ new neighbors with the same color as v, and then put v in V_1. This step, however, can possibly yield more than a quadratic number of vertices. To obtain a quadratic kernel without annotated vertices, we add two new rules.

Rule 17 *Let* $v \in V_0$ *be a vertex with fixed color* c*. Let* w *be a neighbor of* v*, and suppose that the vertex set of the component of* $F - v$ *that contains* w *does not contain a vertex with color* c*, i.e.,* $c \notin S(w)$*. Then remove the edge* $\{v, w\}$ *from* F*.*

Lemma 20 *Rule 17 is sound.*

Proof Let F be the forest before applying the rule, and F' the forest after applying the rule. It is easy to see that each normalized convex recoloring of F that does not recolor vertices in V_0 is a normalized convex recoloring of F' that does not recolor vertices in V_0, and vice verse. □

Proposition 3 *If Rules 8 and 17 cannot be applied, then each vertex in V_0 has degree at most one.*

Proof Suppose $v \in V_0$ has degree at least two. Let T be the tree in F that contains v. If at least two subtrees of $T - v$ contain vertices with the same color as v, then Rule 8 can be applied. Otherwise, there is a subtree of $T - v$ that does not contain a vertex with the same color as v, and the edge from v to that subtree can be removed by Rule 17. □

We now can undo annotations by applying the following steps for each vertex with a fixed color. See Fig. 10 for an example.

- Suppose $v \in V_0$ has color c, and suppose there are α vertices with color c.
- If c is a broken color, then add $k + 1$ new vertices with color c, and make these incident to v.
- If c is not a broken color, then add $\min\{k + 1, \alpha - 1\}$ new vertices with color c, and make these incident to v.
- Unfix the color of v, i.e., put v in V_1.

Lemma 21 *Let F' be obtained from F by undoing the annotation for vertex v as described. Then there is a convex recoloring of F with at most k recolored vertices, not recoloring vertices in V_0, if and only if there is such a convex recoloring of F'.*

Proof Clearly, if there is a convex recoloring Γ' of F with at most k recolored vertices (not recoloring vertices in V_0), there is one of F': we extend Γ' to F' by not recoloring any new vertex.

Now, suppose we have a convex recoloring Γ' of F' with at most k recolored vertices, that does not recolor vertices in V_0. If Γ' does not recolor v, then the restriction of Γ' to F is the requested recoloring of F. So suppose v is recolored by Γ'. We can assume Γ' is normalized. If we added $k + 1$ neighbors with color c, then by Lemma 19, v is not recolored. Suppose c is not a broken color, and we added $\alpha - 1 < k + 1$ neighbors of v with color c. As v is recolored, we must have recolored at least $\alpha - 2$ of the new neighbors of v. Now, change Γ' as follows. For each w with $\Gamma'(w) = c$, give w the color c_w. As c was not broken, and Γ' is normalized, these vertices had color c in the original coloring, so we recolor here at most $\alpha - 1$ vertices. Then, give v and all its new neighbors color c: these are at least $\alpha - 1$ vertices that are no longer recolored. We obtain a convex recoloring Γ'' that recolors at most as many vertices as Γ', and Γ'' does not recolor v. Again, we can use the restriction of Γ'' to F. □

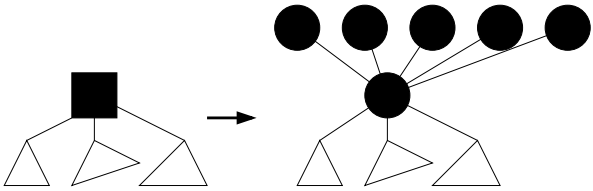

Fig. 10 Example: undo annotations

So, the steps described above allow us to undo all annotations. As there are $O(k)$ broken colors, we add $O(k^2)$ new vertices with a broken color, and for all other colors, we add at most as many new vertices as there were old vertices with such a color. Thus, we still have a kernel with $O(k^2)$ vertices.

With one simple final step, we can turn the forest into a tree. Take $k+2$ new vertices $z_0, z_1, \ldots, z_{k+1}$ with a new color c^*. Make $z_1, \ldots, z_{k+1}$ adjacent to z_0, and make z_0 adjacent to an arbitrary vertex in each tree in F. By Lemma 19, we cannot recolor z_0, and thus, the step gives an equivalent instance with a tree instead of a forest.

Theorem 3 *A kernel for* CONVEX RECOLORING *with* $O(k^2)$ *vertices can be obtained in time polynomial in* n *and* k.

6 Discussion and Open Problems

In this paper, we have given a quadratic kernel for the CONVEX RECOLORING problem for trees. An obvious question is, whether CONVEX RECOLORING for trees admits a linear vertex kernel. We remark that good kernelization results for FPT problems often require a sustained and detailed effort to develop and exploit the structure inherent in the parameterization. One might reasonably wish it were otherwise, but such detailed structural development seems to play an essential role in many results of parameterized algorithmics (clearly evident in the motivating example—for the field—of graph minor theory).

A methodological contribution of this paper is in Sect. 5. The strategy of studying *annotated forms of the problem at hand* has played a major role in the relatively new field of FPT kernelization. One cannot complain, since if the only way to the strongest kernelization results (which have practical significance) is *via* annotated problem forms that increase the modeling power, then one has won twice in one throw. This phenomenon is not well-understood. In Sect. 5, we pull the curtain aside, and show that the annotated problem can be translated back to the original problem. It would be interesting if something similar can be achieved for other FPT problems where the best kernelization results presently seem to depend on annotation. Note that this back-translation, while easier than our main result, is still non-trivial.

Other questions for further research include exploring efficient kernelization for natural variants of CONVEX RECOLORING that are motivated by realistic applications. Several such variants have been introduced by Moran and Snir [20, 21] in the setting of phylogenetic problems, and by Chen, Hu and Shuai [9] in the setting of network routing problems, and by Kammer and Tholey [19] who consider relaxed models of routing problems. Interestingly, it cannot be taken for granted that just because

a basic FPT combinatorial problem admits efficient P-time kernelization, then so do its seemingly minor variants. For example, although the FPT VERTEX COVER problem has a well-known $2k$-vertex P-time kernelization due to Nemhauser and Trotter [22], it has recently been shown that the seemingly minor variant (also trivially FPT) CONNECTED VERTEX COVER does not admit any P-time kernelization to a graph on a polynomial number of vertices, unless the Polynomial Hierarchy collapses to the third level [5, 12, 15]. The entire area of FPT kernelization seems to be fraught with deep and difficult questions; albeit, when efficient P-time kernelization can be achieved—with substantial practical payoffs [1, 2, 18].

In the case of the elegantly formulated CONVEX RECOLORING problem addressed here, it is satisfying and reassuring that the initial $Poly(k)$ kernel, due to unpublished work by some of the authors of this paper, (that was based on twelve reduction rules—many different from those employed here) and that had a vertex kernel bound of $O(k^{28})$, is improved here to $O(k^2)$, based on a markedly deeper attention to the structure associated to the parameter.

Acknowledgement We thank the anonymous referees for very helpful comments on the paper.

References

1. Abu-Khzam, F.N., Collins, R.L., Fellows, M.R., Langston, M.A., Suters, W.H., Symons, C.T.: Kernelization algorithms for the vertex cover problem: Theory and experiments. In: Proc. 6th ACM-SIAM ALENEX, pp. 62–69. ACM-SIAM (2004)
2. Alber, J., Betzler, N., Niedermeier, R.: Experiments in data reduction for optimal domination in networks. Ann. Oper. Res. **146**, 105–117 (2006)
3. Alber, J., Fellows, M.R., Niedermeier, R.: Polynomial-time data reduction for dominating sets. J. ACM **51**, 363–384 (2004)
4. Bar-Yehuda, R., Feldman, I., Rawitz, D.: Improved approximation algorithm for convex recoloring of trees. Theory Comput. Syst. **43**, 3–18 (2008)
5. Bodlaender, H.L., Downey, R.G., Fellows, M.R., Hermelin, D.: On problems without polynomial kernels. J. Comput. Syst. Sci. **78**, 160–174 (2009)
6. Bodlaender, H.L., Fellows, M.R., Langston, M., Ragan, M., Rosamond, F., Weyer, M.: Quadratic kernelization for convex recoloring of trees. In: Lin, G. (ed.) Proceedings 13th Annual International Computing and Combinatorics Conference, COCOON 2007, Heidelberg. Lecture Notes in Computer Science, vol. 4598, pp. 86–96. Springer, Berlin (2007)
7. Burrage, K., Estivill-Castro, V., Fellows, M.R., Langston, M.A., Mac, S., Rosamond, F.A.: The undirected feedback vertex set problem has a poly(k) kernel. In: Bodlaender, H.L., Langston, M.A. (eds.) Proceedings 2nd International Workshop on Parameterized and Exact Computation, IWPEC 2006. Lecture Notes in Computer Science, vol. 4169, pp. 192–202. Springer, Berlin (2006)
8. Chen, J., Fernau, H., Kanj, I.A., Xia, G.: Parametric duality and kernelization: Lower bounds and upper bounds on kernel size. SIAM J. Comput. **37**, 1077–1106 (2007)
9. Chen, X., Hu, X., Shuai, T.: Inapproximability and approximability of maximal tree routing and coloring. J. Comb. Optim. **11**, 219–229 (2006)
10. Courcelle, B.: The monadic second-order logic of graphs I: Recognizable sets of finite graphs. Inf. Comput. **85**, 12–75 (1990)
11. Courcelle, B.: Graph grammars, monadic second-order logic and the theory of graph minors. In: Robertson, N., Seymour, P. (eds.) Graph Structure Theory, Proceedings of the AMS-IMS-SIAM Joint Summer Research Conference, Seattle WA, June 1991. Contemp. Math., vol. 147, pp. 565–590. Am. Math. Soc., Providence (1993)

12. Dom, M., Lokshtanov, D., Saurabh, S.: Incompressibility through colors and IDs. In: Albers, S., Marchetti-Spaccamela, A., Matias, Y., Nikoletseas, S.E., Thomas, W. (eds.) Automata, Languages and Programming, Proceedings of the 36th International Colloquium, ICALP 2009, Part I. Lecture Notes in Computer Science, vol. 5555, pp. 378–389. Springer, Berlin (2009)
13. Downey, R.G., Fellows, M.R.: Parameterized Complexity. Springer, Berlin (1999)
14. Flum, J., Grohe, M.: Parameterized Complexity Theory. Springer, Berlin (2006)
15. Fortnow, L., Santhanam, R.: Infeasibility of instance compression and succinct PCPs for NP. In: Proceedings of the 38th Annual Symposium on Theory of Computing, STOC 2006, pp. 133–142. ACM, New York (2008)
16. Gramm, J., Nickelsen, A., Tantau, T.: Fixed-parameter algorithms in phylogenetics. Comput. J. **51**, 79–101 (2008)
17. Guo, J., Niedermeier, R.: Invitation to data reduction and problem kernelization. ACM SIGACT News **38**, 31–45 (2007)
18. Hüffner, F.: Algorithms and experiments for parameterized approaches to hard graph problems. PhD thesis, Friedrich-Schiller University, Jena, Germany (2007)
19. Kammer, F., Tholey, T.: The complexity of minimum convex coloring. In: Hong, S.-H., Nagamochi, H., Fukunaga, T. (eds.) Proceedings 19th International Symposium on Algorithms and Computation, ISAAC 2008. Lecture Notes in Computer Science, vol. 5369, pp. 16–27. Springer, Berlin (2008)
20. Moran, S., Snir, S.: Efficient approximation of convex recolorings. J. Comput. Syst. Sci. **73**, 1078–1089 (2007)
21. Moran, S., Snir, S.: Convex recolorings of strings and trees: Definitions, hardness results and algorithms. J. Comput. Syst. Sci. **74**, 850–869 (2008)
22. Nemhauser, G.L., Trotter, L.E.: Vertex packing: Structural properties and algorithms. Math. Program. **8**, 232–248 (1975)
23. Niedermeier, R.: Invitation to Fixed-Parameter Algorithms. Oxford Lecture Series in Mathematics and Its Applications. Oxford University Press, London (2006)
24. Ponta, O., Hüffner, F., Niedermeier, R.: Speeding up dynamic programming for some NP-hard graph recoloring problems. In: Agrawal, M., Du, D.-Z., Duan, Z., Li, A. (eds.) Proceedings 5th International Conference on Theory and Applications of Models of Computation, TAMC 2008. Lecture Notes in Computer Science, vol. 4978, pp. 490–501. Springer, Berlin (2008)
25. Thomassé, S.: A quadratic kernel for feedback vertex set. In: Proceedings of the 19th Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2009, pp. 115–119 (2009)